Model-Based Testing with Graph Transformation Systems

Vincent de Bruijn

March 5, 2012

Abstract

This report describes the setup of a research project where the use of *Graph Transformation Systems* (GTS) in model-based testing is researched. The goal of the project is to create a system for automatic test generation on GTSs and to validate this system. A graph transformation tool, GROOVE, and a model-based testing tool, ATM, are used as part of the system. The system will be validated using the results of several case-studies. GTSs have many structural advantages, which are potential benefits for the model-based testing process.

Contents

Chapter 1

Introduction

Chapter 2

Design

2.1 GTS to STS transformation rules

2.1.1 Graph state to location

Generalized graph states.

2.1.2 Rule match to switch relation

Rule name to gate.

Variables

Variable node match from rule to graph state is location variable. Parameterized variable nodes are interaction variables.

Guard

NAC expressions over variable nodes in lhs are guards.

Update

NAC expressions over variable nodes in rhs are updates.

2.2 Exploration strategy

A GROOVE exploration strategy finds all graph states and rule matches.

Isomorphism

Isomorphism prevents correct tracking of location variables. Solve with sets.

Eraser/creator pairs

Location variables should be persistent and declared at the start state. Therefore, every eraser edge to a variable node should be paired with a creator edge. This edge should have the same source node and label.

2.3 Partial matching

2.3.1 Theory

Generalizing rule matches.

2.3.2 Application

Using partial matching in an exploration strategy, only the generalized graph states and rule matches are found.

2.4 Reachability

2.4.1 Theory

MSc thesis Floor Sietsma "A Case Study in Formal Testing and an Algorithm for Automatic Test Case Generation with Symbolic Transition Systems"

2.4.2 Application

Allows halting an exploration strategy when location no longer reachable. Only works for specific path though.

2.5 Pre-testing vs. on-the-fly transformation

Pre-testing transformation is the process of entirely transforming a model before starting test runs. *On-the-fly* transformation is the process of transforming the needed parts of a model while doing a test run.

Explore entire sts + coverage statistics vs. on the fly exploration and no coverage statistics. Reachability check in exploration strategy vs. check in on the fly exploration.

Chapter 3

Validation

3.1 Case Studies

3.2 Benchmarks

Chapter 4

Conclusion

4.1 Summary

This report motivates the need of a research towards a model-based testing practice on Graph Transformation Systems. The goal will be to create a tool that allows automatic test generation on GTSs and assess the strengths and weaknesses of this test practice.

The first observations in this report demonstrate that GTSs can provide a nice visiualization of a system. However, the representation of data values and in a GTS were not Increasing the complexity of the software system may change this. The transformation rules, given as separate graphs, provide a good overview of the behavior of the system. This feature should be beneficial in models for larger software systems.

The results also show an interesting automatic statespace reduction, namely the symmetry reduction. The second GTS of the example shows that also a purely arithmetic model can be built; this indicates the strength of the formalism.

The design phase is split into small steps that should break down the complexity of the entire implementation process. The validation with the use of the case studies emphasises the practicality of the tooling; the purpose of the test tools is to be used on real-world software systems. Finally, the experiments are a great indication of the usefulness of the tool towards software testers.

4.2 Conclusion

4.3 Future Work